Bandit-Based Solar Panel Control

David Abel, Edward C. Williams, Stephen Brawner,
Emily Reif, Michael L. Littman
Brown University Computer Science Department
115 Waterman Street
Providence, RI 02912
{david_abel,edward_c_williams,stephen_brawner,emily_reif,michael_littman}@brown.edu

Abstract

Solar panels sustainably harvest energy from the sun. To improve performance, panels are often equipped with a tracking mechanism that computes the sun's position in the sky throughout the day. Based on the tracker's estimate of the sun's location, a controller orients the panel to minimize the angle of incidence between solar radiant energy and the photovoltaic cells on the surface of the panel, increasing total energy harvested. Prior work has developed efficient tracking algorithms that accurately compute the sun's location to facilitate solar tracking and control. However, always pointing a panel directly at the sun does not account for diffuse irradiance in the sky, reflected irradiance from the ground and surrounding surfaces, power required to reorient the panel, shading effects from neighboring panels and foliage, or changing weather conditions (such as clouds), all of which are contributing factors to the total energy harvested by a fleet of solar panels. In this work, we show that a bandit-based approach can increase the total energy harvested by solar panels by learning to dynamically account for such other factors. Our contribution is threefold: (1) the development of a test bed based on typical solar and irradiance models for experimenting with solar panel control using a variety of learning methods, (2) simulated validation that bandit algorithms can effectively learn to control solar panels, and (3) the design and construction of an intelligent solar panel prototype that learns to angle itself using bandit algorithms.

Introduction

Solar energy offers a pollution free and sustainable means of harvesting energy from the sun. Considerable effort has been directed toward maximizing the efficiency of end-to-end solar systems, including the design of photovoltaic cells (Li 2012), engineering new photovoltaic architectures and materials (Li et al. 2005), and solar tracking systems (Camacho and Berenguel 2012). Solar tracking is especially important for maximizing performance of solar panels (2008). Given the proper hardware, a tracking algorithm can compute the relative location of the sun in the sky throughout the day, and a controller can orient the panel to point at the sun. Its goal is to minimize the angle of incidence between incoming solar radiant energy and the grid of photovoltaic cells (King, Boyson, and Kratochvil 2001; Kalogirou 1996).

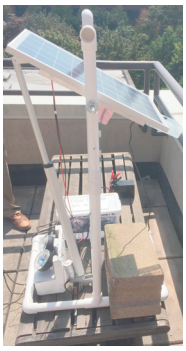


Figure 1: Our intelligent solar panel prototype.

Prior work has consistently demonstrated that panels using a tracking system increase the total energy by a substantial amount: Eke and Senturk (2012) report that a dual-axis tracker yielded 71 kW/h, compared to a fixed panel's yield of 52 kW/h on the same day. They also report energy harvesting gains of dual-axis tracking systems over fixed systems varying from 15% to 40%, depending on the time of year. Mousazadeh et al. (2009) report that gains from tracking can vary between 0% and 100%, while Clifford and Eastwood (2004) report a gain of 23% due to tracking in simulation.

Recent work in solar tracking has focused on developing algorithms that are sufficiently accurate to inform control of panels, building on the early work of Spencer (1971) and Michalsky (1988). The algorithm introduced by Reda and Andreas (2004) computes the sun's location in the sky within $\pm 0.0003^\circ$ of accuracy, achieving the highest degree of accuracy of any known algorithm, but is computationally inefficient to the point of impracticality. Grena (2008) overcomes these inefficiencies with a tracking algorithm that requires an order of magnitude fewer calculations while still achieving 0.0027° of accuracy.

However, prior literature suggests that a variety of factors contribute to the performance of a panel (King, Boyson, and Kratochvil 2001), and thus, pointing a panel directly at the sun is not always optimal behavior. Specifically, the total solar irradiance falling on a panel is a combination of *direct*, *reflective*, and *diffuse* irradiance (Benghanem 2011). The

diffuse irradiance typically varies between 15% and 55% of direct irradiance depending on factors like cloud coverage and the time of day (Peterson and Dirmhirn 1981), while a case study by the Cold Climate Housing Research Center in Fairbanks, Alaska reports reflective irradiance varying from 5% to 25% of direct irradiance (Colgan et al. 2010). The reflective irradiance varies heavily based on the percentage of irradiance reflected off the surrounding ground surface: Typical values for this percentage given by McEvoy et al. (2003) vary between 17% (soil), 25% (grass), 55% (concrete), and 90% (snow). Additionally, changing weather and atmospheric conditions can affect the optimal panel orientation (Kelly and Gibson 2009), as well as shading effects from neighboring panels and foliage (Passias and Källbäck 1984). Thus, optimal performance may involve prioritizing reflective or diffuse irradiance when direct sunlight is not available. Other shortcomings of the classical tracking approach include the need for additional hardware, limited time-periods of accuracy (some trackers are only accurate until 2023, for instance), and ignoring the power cost of reorienting panels.

In this work, we advocate for the use of machine learning to optimize solar panel performance. A learned solar panel controller can account for weather change, cloud coverage, power use, shading effects, and diverse reflective indices of surroundings, offering an efficient yet adaptable solution that can optimize for the given availability of each type of solar irradiance regardless of the location or year. In particular, we take the non-sequential contextual bandit problem to accurately model the problem of orienting solar panels to maximize energy gathered. We also explore the use of a Reinforcement Learning (RL) approach to gain insight into the advantage of modeling the sequential movements of the sun, panel, and weather.

Our contribution is threefold:

1. The advancement of a relevant problem as an application area for bandits and RL, including a simulation for evaluating learning methods.
2. The validation of the utility of a contextual bandit approach for solar panel control in simulation.
3. The design and construction of an intelligent solar panel prototype, which learns to orient itself over time to maximize energy.

Background

First, some background on solar tracking, bandits, and RL.

Solar Tracking

The amount of solar radiant energy contacting a surface on the earth's surface (per unit area, per unit time) is called *irradiance* (Goswami, Kreith, and Kreider 2000). We denote the total irradiance hitting a panel as R_t, which, per the model developed by Kamali, Moradi, and Khalili (2006), is approximated by the sum of the *direct* irradiance, R_d, *diffuse* irradiance (light from the sky), R_f, and *reflective* irradiance, R_r (reflected off the ground or other surfaces). Each of these components is modified by a scalar, $\theta_d, \theta_f, \theta_r \in [0, 1]$, denoting the effect of the angle of incidence between oncoming solar rays and the panel's orientation, yielding the total:

$$R_t = R_d\theta_d + R_f\theta_f + R_r\theta_r. \quad (1)$$

Additionally, the components R_d and R_f are known to be effected by cloud coverage (Li, Lau, and Lam 2004; Pfister et al. 2003; Tzoumanikas et al. 2016). There is little consensus regarding a precise model of *how* cloud coverage effects these values, in part due to the high variance in cloud composition and coverage. We anticipate that adverse weather conditions can only help improve the learner's performance relative to existing baselines, since classical panel controllers exhibit the same behavior regardless of weather.

A controller for a solar panel seeks to maximize total irradiance R_t hitting the panel's surface. Solar tracker controllers orient the panel so that its normal vector is pointing at the sun. For an in depth survey of solar tracking techniques, see the work of Mousazadeh et al. (2009).

Bandits

We model the problem of maximizing a solar panel's energy intake as a non-sequential *contextual bandit* problem introduced by Wang, Kulkarni, and Poor (2005). This setting extends the classic multi-armed bandit problem (Gittins 1979) to include a *context matrix* $\boldsymbol{X}$ containing a feature vector for each action. That is, each column of $\boldsymbol{X}$ corresponds to each action's context: the entry $\boldsymbol{X}_{i,j}$ denotes the i-th feature of action a_j. We let $\boldsymbol{x}^a$ denote the context vector associated with action a. At each time step, the agent chooses an action $a \in \mathcal{A}$, and receives payoff according to an unknown, possibly stochastic function, $\mathcal{R}(\boldsymbol{x}, a)$. The agent's goal is to maximize total payoff over the course of its action choices. Here, in addition to the typical learning problem of induction, agents face the exploration–exploitation dilemma (but do not need to learn their estimates from delayed reward as is the case in RL). The context vector for each action is the received percept from the environment. (There is no difference in contexts across actions for the solar problem.)

The contextual bandit is an appropriate model for the problem of maximizing solar radiant energy since decisions made at each time step are largely independent of one another: energy received by the panel *primarily* depends on the most immediate action taken by the agent, not by a long history of actions.

Reinforcement Learning

To explore the effects of modeling the environment as sequential, we also experimented with RL, a computational learning paradigm in which an agent learns to make decisions that maximize an unknown reward function through repeated interaction with the agent's environment. RL usually models the environment as a Markov Decision Process or MDP (Puterman 2014), which is a five tuple, $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma \rangle$, where:

1. $\mathcal{S}$ is a set of states,
2. $\mathcal{A}$ is a set of actions,
3. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \longmapsto \mathbb{R}$ is a reward function,

4. $\mathcal{T}(s' \mid s, a)$ is a probability distribution on next states given a state and action,
5. $\gamma \in [0, 1)$ is a discount factor, indicating how much the agent prefers immediate reward over future reward.

The solution to an MDP is called a *policy*, denoted $\pi : \mathcal{S} \longmapsto \mathcal{A}$. Similarly to the bandits problem, the agent's goal is to solve for a policy that maximizes long term expected reward, defined by the *value function*, $V^* : S \longmapsto \left[0, \frac{\text{RMAX}}{1-\gamma}\right]$, given by the classic Bellman Equation:

$$V^*(s) = \max_a \left(\mathcal{R}(s, a) + \gamma \sum_{s'} \mathcal{T}(s' \mid s, a) V^*(s') \right).$$

Also of interest is the *action-value function*, $Q^* : \mathcal{S} \times \mathcal{A} \longmapsto \mathbb{R}$, which specifies the long term expected reward of executing an action in a state and behaving optimally thereafter:

$$Q^*(s, a) = \mathcal{R}(s, a) + \gamma \sum_{s'} \mathcal{T}(s' \mid s, a) V^*(s'). \quad (2)$$

For further background on RL, see the work of Sutton and Barto (1998) and Kaelbling, Littman, and Moore (1996).

Learning Algorithms

Due to its simplicity and efficiency, we experiment with the LinUCB algorithm developed by Li et al. (2010). LinUCB adapts the core ideas of the UCB (Upper Confidence Bound) algorithm (Auer, Cesa-Bianchi, and Fischer 2002) to deal with contexts. At a high level, LinUCB assumes the underlying reward is determined by a linear payoff matrix $\boldsymbol{\theta}$ and maintains a running estimate $\hat{\boldsymbol{\theta}}$. The critical piece of the algorithm is its exploration strategy, which calculates a confidence interval on the difference between the agent's estimate of the expected reward and the actual return, which is factored into an overall *score* for each action. At each round, the agent selects the action with max score according to:

$$\pi(s) = \arg\max_{a \in \mathcal{A}} \left(\sum_i \boldsymbol{x}_i^a \hat{\boldsymbol{w}}_i^a + \sigma^a \right), \quad (3)$$

where σ^a represents the confidence interval associated with action a.

SARSA (Rummery and Niranjan 1994) maintains an estimate of Q^* via updates after each experience $\langle s, a, r, s', a' \rangle$, updating according to the rule:

$$\hat{Q}(s, a) = (1 - \eta)\hat{Q}(s, a) + \eta(r + \gamma \hat{Q}(s', a')), \quad (4)$$

where $\eta \in [0, 1]$ is a learning rate. The linear approximator extends tabular SARSA to domains where states are described by feature vectors, $\boldsymbol{s} = [s_1 \quad s_2 \quad \ldots \quad s_k]$. Here, $\hat{Q}$ is parameterized by a set of k-vectors w^a, where each vector corresponds to action a's parameters across the state variables. We pair SARSA with a typical ϵ-greedy policy.

We chose these two algorithms to illustrate that online, efficient, and lightweight algorithms can be effective in the solar domain. We chose not to experiment with any Deep RL approaches like the DQN (Mnih et al. 2015), as Deep RL typically requires more computational power (and often GPUs, which expend more energy), which may be unavailable or limited in our setting. Evaluating the relative data and energy efficiency is a topic for further exploration.

Simulated Experiments

We introduce a simulated environment to validate the use of learning algorithms for solar panel control. There are four stages to the simulation (1) Computing the sun's location in the sky, relative to the panel, (2) Computing R_d, R_f, and R_r, (3) Computing θ_d, θ_f, and θ_r, (4) Generating percepts.

The solar panel control problem is modeled by an MDP where the energy received at each timestep defines the reward and the actions change the panel's angle. We allow for two types of state description:

1. **simple** Four variables describing the orientation of the panel and the angles describing the relative location of the sun in the sky.
2. **image**: 256 variables denoting greyscale pixel intensities of a 16x16 synthesized image of the sky with cloud cover. Example images appear in Figure 2.

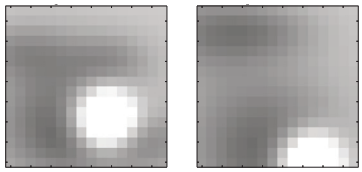


Figure 2: Example images given to the learning agents.

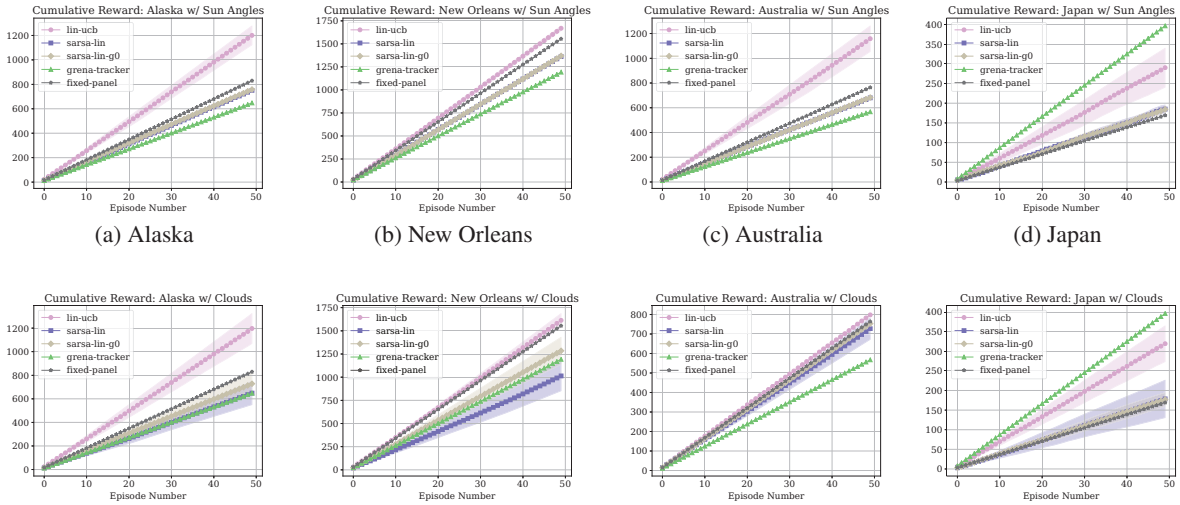
The learning agents have two different action spaces depending on the number of axes of freedom offered to the panel and the type of agent. In the single axis case, SARSA has three actions: tilt forward, tilt back, and do nothing. LinUCB has one action for each possible discrete orientation of the panel. In the dual axis case, the RL agent has two extra forward and back actions that allow the panel to rotate along the other axis, and the bandit algorithm has an action for each possible orientation along *both* axes.

We set each panel action to move the panel 5° in the specified direction, where the agents take an action every five minutes of simulated wall clock time. We set the reflective index to 0.55 (the index of concrete), the uncertainty parameter of LinUCB to 2.0, and ϵ and η to 0.1 and 0.05 respectively for SARSA. We used an annealing schedule where:

$$\eta_t = \max\left\{(0.0001 * t * \eta_0)^2, 0\right\},$$
$$\epsilon_t = \max\left\{(0.0001 * t * \epsilon_0)^2, 0\right\},$$

with t denoting the current timestep of the agent's learning process. For more details on the simulation, see earlier versions of this work (Abel, Reif, and Littman 2017; Abel et al. 2017).

We primarily experimented with a single-axis panel in simulation to parallel our prototype. We compare the effectiveness of a traditional solar tracker—Algorithm 2 of Grena (2012)—(`grena-tracker`), the contextual bandit



(a) Alaska (b) New Orleans (c) Australia (d) Japan

(e) Alaska (f) New Orleans (g) Australia (h) Japan

Figure 3: Cumulative reward for simulated experiments for the simple percept (top) and image percept (bottom).

approach (`lin-ucb`), two RL approaches (`sarsa-lin`, `sarsa-lin-`γ_0), and a fixed tracker (`fixed`).

We ran each simulated experiment in four different locations on Earth in 2020: Juno AK, New Orleans LA, Sydney Australia, and Tokyo Japan. We ran each agent in an episodic fashion; each episode consisted of the same 24 hours of daylight (skipping nighttime). We repeated each experiment 10 times and report 95% confidence intervals. All of our code for reproducing results (and continued experimentation with the simulation) is publicly available.[1]

Simulated Results

Results for the single axis experiments with simple percepts are shown in the top row of Figure 3. Notably, `lin-ucb` performs competitively with the baselines across the board. In Australia, Alaska, and New Orleans, we see `lin-ucb` improve over all other approaches. In Japan, after a number of episodes, the policy found by `lin-ucb` improves substantially and separates itself from the fixed panel and RL agents but still performs worse than `grena`. Further, we note there is little distinction between myopic SARSA (with $\gamma = 0$) and sequential SARSA, suggesting that, at least in simulation, modeling the problem as sequential is not useful.

The results of the second experiment (with image percepts) are shown in the bottom row of Figure 3. The algorithms perform comparably to the prior experiment, suggesting that the added complexity of the image percept doesn't change the difficulty of the learning problem. Further, we note that, in Japan, LinUCB actually perform better with the image percepts.

To better diagnose results, we illustrate the ratio of each energy type acquired. The table in Figure 4 summarizes average energy accumulated of each type in New Orleans for the simple percept experiment, which is representative of ratios in other experiments. The advantage of `lin-ucb` comes from both direct and reflective energy—this is due to the fact that the `grena` tracker optimizes for a proxy criteria: minimizing the distance between the panel's normal vector and the angle of the sun. Conversely, the learner acts so as to maximize the cumulative energy directly, better enabling it to take advantage of available energy.

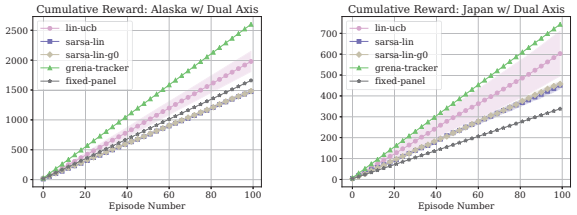
We also conducted proof-of-concept experiments for the dual axis system in Alaska and Japan. The results are shown in Figure 5. We increased the panel angle of movement from $5°$ to $20°$ to limit LinUCB's action space. The `grena-tracker` consistently outperformed the `fixed-panel` in all dual-axis experiments by anywhere from 30% to 100% (including results not shown). This reinforces the claim that tracking helps most consistently when two axes of movement are available (contrasted with the single-axis experiments where the `grena-tracker` occasionally performs worse than the `fixed-panel`). Despite the increase in learning difficulty, `lin-ucb` learns a policy that is not too much worse than the `grena-tracker` even with a small sample budget.

The simulation is useful for guiding algorithmic design but does not faithfully model every detail of the real world problem. Most notably, the effect of weather on cumulative irradiance, which is significant, is absent from the simula-

Approach	Direct	Diffuse	Reflective
`lin-ucb`	1382	102	175
`sarsa`	1182	115	73
`grena`	975	112	105

Figure 4: Energy breakdown in New Orleans.

[1] https://github.com/david-abel/solar_panels_rl



(a) Alaska (b) Japan

Figure 5: Cumulative reward for dual axis methods in selected locations.

tion. Since learning methods can dynamically accommodate a variety of circumstances presented by different weather patterns, foliage, air pollution, and shading effects, we expect learning approaches to improve. Conversely, the classical tracking approaches will execute the same control regardless of changing weather conditions. Thus, we expect improvements on the prototype to be even larger (and more consistent across locations) than those in simulation.

Prototype

We constructed a physical prototype of a learning solar panel, shown in Figure 1. The prototype is a simple single-axis solar tracker with a linear actuator rotating the vertically mounted panel along the north-south axis. The 18-inch actuator orients the panel by extending up and down, enabling nearly the full 180° degrees of motion. Learning algorithms and baselines were executed on a Raspberry Pi single-board computer, which relayed commands to and received energy data from an Arduino microcontroller. The panel was connected to a solar charge controller and a 12V battery, which powered the control electronics. The panel was set to move in intervals of 0.1 radians (5.8°), as measured by an accelerometer mounted on the panel itself. Each agent waited for one minute to collect reward data before taking another action, and agents were switched every thirty minutes. Visual percepts were acquired via a USB webcam whose output was downsampled to 16x16 grayscale images.

We evaluated our learning approaches against baselines on the roof of the Brown Center for Information Technology in Providence, RI over the course of a week, wherein each control approach got blocks of time to control the panel during any given day. Naturally, real world data is more time consuming to collect than in simulation. Additionally, iterating on and improving the reliability of our control electronics required interruption of the learning process for repairs and upgrades. As a result, we were only able to collect data over the course of several hours of continuous learning per agent, rather than the 24 hours in the simulated experiment. This rendered agents vulnerable to behavior spikes due to brief intervals of cloud cover or shading from tall buildings near the panel.

Energy gathered by each agent over the course of a three-hour period on September 9, 2017 is displayed in Figure 7. The agent controlled by `lin-ucb` handily beats the other agents, but plotting the results per timestep as in Figure 6

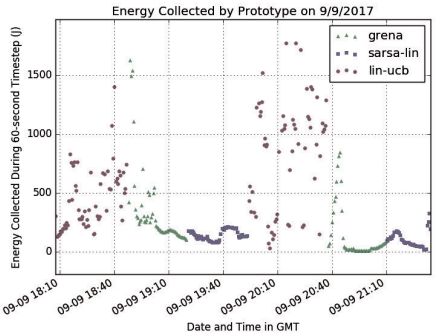


Figure 6: Energy collected per 1-minute timestep by each approach over the course of a day controlling the panel.

Approach	Total Energy Gathered (J)
`lin-ucb`	77103.19
`sarsa`	12219.55
`grena-tracker`	26600.33

Figure 7: Total energy collected.

shows the effects of limited observation time and environmental change on results. The large variance in `lin-ucb` performance during an observation period shows the agent exploring different panel positions. However, the similarity in performance between the `grena` tracker and `sarsa` likely indicates that both agents were affected by cloud cover or other shading.

The data collected is limited enough that no valid conclusions can be drawn—however, we note that `lin-ucb` performs comparably to the fixed-policy `grena` tracker in shade-free observation periods. A natural next step is to continue data collection, extending learning intervals into days for each agent. We also plan to modify our single-axis tracker to allow rotational motion across a second axis, giving it a greater ability to track the sun while simultaneously increasing the size (and difficulty) of the posed learning problem.

Conclusion

We demonstrated the benefits of using a learning approach to improve the efficiency of solar panels over established baselines. We introduced a simulation to test learning approaches for solar energy harvesting with solar panels. Lastly, we described the design and implementation of a functioning RL controlled solar panel, with preliminary findings about its performance with learning algorithms.

References

Abel, D.; Reif, E.; Williams, E. C.; and Littman, M. L. 2017. Toward improving solar panel efficiency using reinforcement learning. *EnviroInfo 2017*.

Abel, D.; Reif, E.; and Littman, M. L. 2017. Improving

solar panel efficiency using reinforcement learning. *RLDM 2017*.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47(2-3):235–256.

Benghanem, M. 2011. Optimization of tilt angle for solar panel: Case study for Madinah, Saudi Arabia. *Applied Energy* 88(4):1427–1433.

Camacho, E. F., and Berenguel, M. 2012. Control of solar energy systems1. *IFAC Proceedings Volumes* 45(15):848–855.

Clifford, M., and Eastwood, D. 2004. Design of a novel passive solar tracker. *Solar Energy* 77(3):269–280.

Colgan, R.; Wiltse, N.; Lilly, M.; LaRue, B.; and Egan, G. 2010. Performance of photovoltaic arrays. *Cold Climate Housing Research Center*.

Eke, R., and Senturk, A. 2012. Performance comparison of a double-axis sun tracking versus fixed PV system. *Solar Energy* 86(9):2665–2672.

Gittins, J. C. 1979. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)* 148–177.

Goswami, D. Y.; Kreith, F.; and Kreider, J. F. 2000. *Principles of solar engineering*. CRC Press.

Grena, R. 2008. An algorithm for the computation of the solar position. *Solar Energy* 82(5):462–470.

Grena, R. 2012. Five new algorithms for the computation of sun position from 2010 to 2110. *Solar Energy* 86(5):1323–1337.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4:237–285.

Kalogirou, S. A. 1996. Design and construction of a one-axis sun-tracking system. *Solar Energy* 57(6):465–469.

Kamali, G. A.; Moradi, I.; and Khalili, A. 2006. Estimating solar radiation on tilted surfaces with various orientations: a study case in karaj (iran). *Theoretical and applied climatology* 84(4):235–241.

Kelly, N. A., and Gibson, T. L. 2009. Improved photovoltaic energy output for cloudy conditions with a solar tracking system. *Solar Energy* 83(11):2092–2102.

King, D. L.; Boyson, W. E.; and Kratochvil, J. A. 2001. Analysis of factors influencing the annual energy production of photovoltaic systems. *Conference Record of the Twenty-Ninth IEEE Photovoltaic Specialists Conference, 2002*. 1356–1361.

Li, G.; Shrotriya, V.; Huang, J.; Yao, Y.; Moriarty, T.; Emery, K.; and Yang, Y. 2005. High-efficiency solution processable polymer photovoltaic cells by self-organization of polymer blends. *Nature materials* 4(11):864–868.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, 661–670. ACM.

Li, D. H.; Lau, C. C.; and Lam, J. C. 2004. Overcast sky conditions and luminance distribution in hong kong. *Building and Environment* 39(1):101–108.

Li, Y. 2012. Molecular design of photovoltaic materials for polymer solar cells: toward suitable electronic energy levels and broad absorption. *Accounts of Chemical Research* 45(5):723–733.

McEvoy, A.; Markvart, T.; Castañer, L.; Markvart, T.; and Castaner, L. 2003. *Practical handbook of photovoltaics: fundamentals and applications*. Elsevier.

Michalsky, J. J. 1988. The astronomical almanac's algorithm for approximate solar position (1950–2050). *Solar energy* 40(3):227–235.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Mousazadeh, H.; Keyhani, A.; Javadi, A.; Mobli, H.; Abrinia, K.; and Sharifi, A. 2009. A review of principle and sun-tracking methods for maximizing solar systems output. *Renewable and sustainable energy reviews* 13(8):1800–1818.

Passias, D., and Källbäck, B. 1984. Shading effects in rows of solar cell panels. *Solar Cells* 11(3):281–291.

Peterson, W. A., and Dirmhirn, I. 1981. The ratio of diffuse to direct solar irradiance (perpendicular to the sun's rays) with clear skiesa conserved quantity throughout the day. *Journal of Applied Meteorology* 20(7):826–828.

Pfister, G.; McKenzie, R.; Liley, J.; Thomas, A.; Forgan, B.; and Long, C. N. 2003. Cloud coverage based on all-sky imaging and its impact on surface solar irradiance. *Journal of Applied Meteorology* 42(10):1421–1434.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Reda, I., and Andreas, A. 2004. Solar position algorithm for solar radiation applications. *Solar energy* 76(5):577–589.

Rizk, J., and Chaiko, Y. 2008. Solar Tracking System: More Efficient Use of Solar Panels. *Proceedings of World Academy of Science: Engineering & Technology* 43:313–315.

Rummery, G. A., and Niranjan, M. 1994. *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering.

Spencer, J. 1971. Fourier series representation of the position of the sun. *Search* 2(5):172–172.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Tzoumanikas, P.; Nikitidou, E.; Bais, A.; and Kazantzidis, A. 2016. The effect of clouds on surface solar irradiance, based on data from an all-sky imaging system. *Renewable Energy* 95:314–322.

Wang, C.-C.; Kulkarni, S. R.; and Poor, H. V. 2005. Bandit problems with side observations. *IEEE Transactions on Automatic Control* 50(3):338–355.